Scientific Computing: Eigen and Singular Values

Aleksandar Donev
Courant Institute, NYU[1]
donev@courant.nyu.edu

[1]Course MATH-GA.2043 or CSCI-GA.2112, Spring 2012

February 23rd, 2011

Outline

1. Review of Linear Algebra
2. Eigenvalue Problems
3. Singular Value Decomposition
4. Principal Component Analysis (PCA)
5. Conclusions

Eigenvalue Decomposition

- For a square matrix $\mathbf{A} \in \mathbb{C}^{n \times n}$, there exists at least one λ such that

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{x} \quad \Rightarrow \quad (\mathbf{A} - \lambda \mathbf{I})\,\mathbf{x} = \mathbf{0}$$

- Putting the **eigenvectors** $\mathbf{x}_j$ as columns in a matrix $\mathbf{X}$, and the **eigenvalues** λ_j on the diagonal of a diagonal matrix $\mathbf{\Lambda}$, we get

$$\mathbf{A}\mathbf{X} = \mathbf{X}\mathbf{\Lambda}.$$

- A matrix is **non-defective** or **diagonalizable** if there exist n **linearly independent eigenvectors**, i.e., if the matrix $\mathbf{X}$ is invertible:

$$\mathbf{X}^{-1}\mathbf{A}\mathbf{X} = \mathbf{\Lambda}$$

leading to the **eigen-decomposition** of the matrix

$$\mathbf{A} = \mathbf{X}\mathbf{\Lambda}\mathbf{X}^{-1}.$$

Unitarily Diagonalizable Matrices

- A **unitary** or **orthogonal** matrix $\mathbf{U}$ has orthogonal colums each of which has unit L_2 norm:

$$\mathbf{U}^{-1} = \mathbf{U}^{\star}.$$

  Unitary is used for complex matrices and is more general than orthogonal, reserved for real matrices.
  Recall that star denotes **adjoint** (conjugate transpose).
- Unitary matrices are important because they are **always well-conditioned**, $\kappa_2(\mathbf{U}) = 1$.
- A matrix is **unitarily diagonalizable** if there exist n linearly independent **orthogonal eigenvectors**, $\mathbf{X} \equiv \mathbf{U}$,

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\star}.$$

- Theorem: **Hermitian matrices**, $\mathbf{A}^{\star} = \mathbf{A}$, are unitarily diagonalizable and have **real eigenvalues**.
  For real matrices we use the term **symmetric**.

Non-diagonalizable Matrices

- For matrices that are not diagonalizable, one can use **Jordan form factorizations**, or, more relevant to numerical mathematics, the **Schur factorization** (decomposition):

$$\mathbf{A} = \mathbf{U}\mathbf{T}\mathbf{U}^{\star},$$

  where $\mathbf{T}$ is **upper-triangular** (unlike **Jordan form** where only nonzeros are on super-diagonal).
- The eigenvalues are on the diagonal of $\mathbf{T}$, and in fact if $\mathbf{A}$ is unitarily diagonalizable then $\mathbf{T} \equiv \mathbf{\Lambda}$.
- An important property / use of eigenvalues:

$$\mathbf{A}^n = (\mathbf{U}\mathbf{T}\mathbf{U}^{\star})(\mathbf{U}\mathbf{T}\mathbf{U}^{\star})\cdots(\mathbf{U}\mathbf{T}\mathbf{U}^{\star}) = \mathbf{U}\mathbf{T}(\mathbf{U}^{\star}\mathbf{U})\mathbf{T}(\mathbf{U}^{\star}\mathbf{U})\cdots\mathbf{T}\mathbf{U}^{\star}$$

$$\mathbf{A}^n = \mathbf{U}\mathbf{T}^n\mathbf{U}^{\star}$$

Formal definition of the SVD

Every matrix has a **singular value decomposition**

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\star} = \sum_{i=1}^{p} \sigma_i \mathbf{u}_i \mathbf{v}_i^{\star}$$

$$[m \times n] = [m \times m]\,[m \times n]\,[n \times n]\,,$$

where $\mathbf{U}$ and $\mathbf{V}$ are **unitary matrices** whose columns are the left, $\mathbf{u}_i$, and the right, $\mathbf{v}_i$, **singular vectors**, and

$$\mathbf{\Sigma} = \mathrm{Diag}\left\{\sigma_1, \sigma_2, \ldots, \sigma_p\right\}$$

is a **diagonal matrix** with real positive diagonal entries called **singular values** of the matrix

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0,$$

and $p = \min(m, n)$ is the maximum possible rank of the matrix.

Relation to Eigenvalues

- For **Hermitian (symmetric) matrices**, there is **no fundamental difference** between the SVD and eigenvalue decompositions.
- The squared singular values are **eigenvalues of the normal matrix**:

$$\sigma_i(\mathbf{A}) = \sqrt{\lambda_i(\mathbf{A}\mathbf{A}^\star)} = \sqrt{\lambda_i(\mathbf{A}^\star\mathbf{A})}$$

since

$$\mathbf{A}^\star\mathbf{A} = (\mathbf{V}\mathbf{\Sigma}\mathbf{U}^\star)(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\star) = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^\star$$

- Similarly, the singular vectors are the corresponding eigenvectors up to a sign.

Rank-Revealing Properties

- Assume the rank of the matrix is r, that is, the dimension of the range of $\mathbf{A}$ is r and the dimension of the null-space of $\mathbf{A}$ is $n - r$ (recall the fundamental theorem of linear algebra).
- The SVD is a **rank-revealing** matrix factorization because only r of the singular values are nonzero,

$$\sigma_{r+1} = \cdots = \sigma_p = 0.$$

- The left singular vectors $\{\mathbf{u}_1, \ldots, \mathbf{u}_r\}$ form an **orthonormal basis for the range** (column space, or image) of $\mathbf{A}$.
- The right singular vectors $\{\mathbf{v}_{r+1}, \ldots, \mathbf{v}_n\}$ form an **orthonormal basis for the null-space** (kernel) of $\mathbf{A}$.

The matrix pseudo-inverse

- For square non-singular systems, $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$. Can we generalize the matrix inverse to non-square or rank-deficient matrices?
- Yes: **matrix pseudo-inverse** (Moore-Penrose inverse):

$$\mathbf{A}^{\dagger} = \mathbf{V}\mathbf{\Sigma}^{\dagger}\mathbf{U}^{\star},$$

where

$$\mathbf{\Sigma}^{\dagger} = \mathrm{Diag}\left\{\sigma_1^{-1}, \sigma_2^{-1}, \ldots, \sigma_r^{-1}, 0, \ldots, 0\right\}.$$

- In numerical computations very small singular values should be considered to be zero (see homework).
- The least-squares solution to over- or under-determined linear systems $\mathbf{Ax} = \mathbf{b}$ can be obtained from:

$$\mathbf{x} = \mathbf{A}^{\dagger}\mathbf{b}.$$

Sensitivity of Eigenvalues

- Now consider a perturbation of a diagonalizable matrix $\delta\mathbf{A}$ and see how perturbed the similar matrix becomes:

$$\mathbf{X}^{-1}\left(\mathbf{A}+\delta\mathbf{A}\right)\mathbf{X}=\mathbf{\Lambda}+\delta\mathbf{\Lambda} \quad \Rightarrow$$

$$\delta\mathbf{\Lambda}=\mathbf{X}^{-1}\left(\delta\mathbf{A}\right)\mathbf{X} \quad \Rightarrow$$

$$\|\delta\mathbf{\Lambda}\| \leq \left\|\mathbf{X}^{-1}\right\| \|\delta\mathbf{A}\| \|\mathbf{X}\| = \kappa\left(\mathbf{X}\right)\|\delta\mathbf{A}\|$$

- Conclusion: The conditioning of the eigenvalue problem is related to the **conditioning of the matrix of eigenvectors**.

Conditioning of Eigenvalue problems

$$\|\delta \mathbf{\Lambda}\| \leq \kappa(\mathbf{X}) \|\delta \mathbf{A}\|$$

- If $\mathbf{X}$ is unitary then $\|\mathbf{X}\|_2 = 1$ (from now on we exclusively work with the 2-norm): **Unitarily diagonalizable matrices are always perfectly conditioned!**
- Warning: The **absolute error** in all eigenvalues is of the same order, meaning that the **relative error will be very large** for the smallest eigenvalues.
- The conditioning number for computing eigenvectors is inversely proportional to the **separation between the eigenvalues**

$$\kappa(\mathbf{x}, \mathbf{A}) = \left(\min_j |\lambda - \lambda_j| \right)^{-1}.$$

The need for iterative algorithms

- The eigenvalues are roots of the **characteristic polynomial** of **A**, which is generally of order n.
- According to Abel's theorem, there is no closed-form (rational) solution for $n \geq 5$.
  All eigenvalue algorithms must be iterative!
- There is an important distinction between iterative methods to:
  - Compute **all eigenvalues** (similarity transformations). These are based on dense-matrix factorizations such as the QR factorization, with total cost $O(n^3)$.
  - Compute **only one or a few eigenvalues**, typically the smallest or the largest one (e.g., power method). These are similar to iterative methods for solving linear systems.

Eigenvalues in MATLAB

- In MATLAB, sophisticated variants of the **QR algorithm** (LAPACK library) are implemented in the function *eig*:

$$\Lambda = eig(A)$$

$$[X, \Lambda] = eig(A)$$

- For large or sparse matrices, iterative methods based on the **Arnoldi iteration** (ARPACK library), can be used to obtain a few of the largest eigenvalues:

$$\Lambda = eigs(A, n_{eigs})$$

$$[X, \Lambda] = eigs(A, n_{eigs})$$

- The **Schur decomposition** is provided by $[U, T] = schur(A)$.

Sensitivity (conditioning) of the SVD

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\star$$

- Since unitary matrices have unit 2-norm,

$$\|\delta\Sigma\|_2 \approx \|\delta A\|_2 .$$

- The SVD computation is always **perfectly well-conditioned**!
- However, this refers to absolute errors: The **relative error** of small singular values will be large.
- The **power of the SVD** lies in the fact that it always exists and can be computed stably...but it is somewhat **expensive to compute**.

Computing the SVD

- The SVD can be computed by performing an eigenvalue computation for the **normal matrix $\mathbf{A}^\star\mathbf{A}$** (a positive-semidefinite matrix).
- This squares the condition number for small singular values and is **not numerically-stable**.
- Instead, modern algorithms use an algorithm based on computing eigenvalues / eigenvectors using the QR factorization.
- The cost of the calculation is $\sim O(mn^2)$, of the same order as eigenvalue calculation if $m \sim n$.

Reduced SVD

The **full (standard) SVD**

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\star = \sum_{i=1}^{p} \sigma_i \mathbf{u}_i \mathbf{v}_i^\star$$
$$[m \times n] = [m \times m]\,[m \times n]\,[n \times n]\,,$$

is in practice often computed in **reduced (economy) SVD** form, where $\mathbf{\Sigma}$ is $[p \times p]$:

$$[m \times n] = [m \times n]\,[n \times n]\,[n \times n] \quad \text{for} \quad m > n$$
$$[m \times n] = [m \times m]\,[m \times m]\,[m \times n] \quad \text{for} \quad n > m$$

This contains all the information as the full SVD but can be **cheaper to compute** if $m \gg n$ or $m \ll n$.

In MATLAB

- $[U, \Sigma, V] = svd(A)$ for **full SVD**, computed using a QR-like method.
- $[U, \Sigma, V] = svd(A, 'econ')$ for **economy SVD**.
- The **least-squares solution** for square, overdetermined, underdetermined, or even rank-defficient systems can be computed using *svd* or *pinv* (pseudo-inverse, see homework).
- The q largest singular values and corresponding approximation can be computed efficiently for **sparse matrices** using

$$[U, \Sigma, V] = svds(A, q).$$

Low-rank approximations

- The SVD is a decomposition into **rank-1 outer product matrices**:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\star = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^\star = \sum_{i=1}^{r} \mathbf{A}_i$$

- The rank-1 components $\mathbf{A}_i$ are called **principal components**, the most important ones corresponding to the larger σ_i.
- Ignoring all singular values/vectors except the first q, we get a **low-rank approximation**:

$$\mathbf{A} \approx \hat{\mathbf{A}}_q = \mathbf{U}_q \mathbf{\Sigma}_q \mathbf{V}_q^\star = \sum_{i=1}^{q} \sigma_i \mathbf{u}_i \mathbf{v}_i^\star.$$

- Theorem: This is the **best approximation** of rank-q in the Euclidian and Frobenius norm:

$$\left\| \mathbf{A} - \hat{\mathbf{A}}_q \right\|_2 = \sigma_{q+1}$$

Applications of SVD/PCA

- **Statistical analysis** (e.g., DNA microarray analysis, clustering).
- Data **compression** (e.g., image compression, explained next).
- **Feature extraction**, e.g., face or character recognition (see Eigenfaces on Wikipedia).
- **Latent semantic indexing** for context-sensitive searching (see Wikipedia).
- **Noise reduction** (e.g., weather prediction).
- One example concerning language analysis given in homework.

Image Compression

```
>> A=rgb2gray(imread('basket.jpg'));
>> imshow(A);
>> [U,S,V]=svd(double(A));
>> r=25; % Rank-r approximation
>> Acomp=U(:,1:r)*S(1:r,1:r)*(V(:,1:r))';
>> imshow(uint8(Acomp));
```

Compressing an image of a basket

We used only 25 out of the ~ 400 singular values to construct a rank 25 approximation:

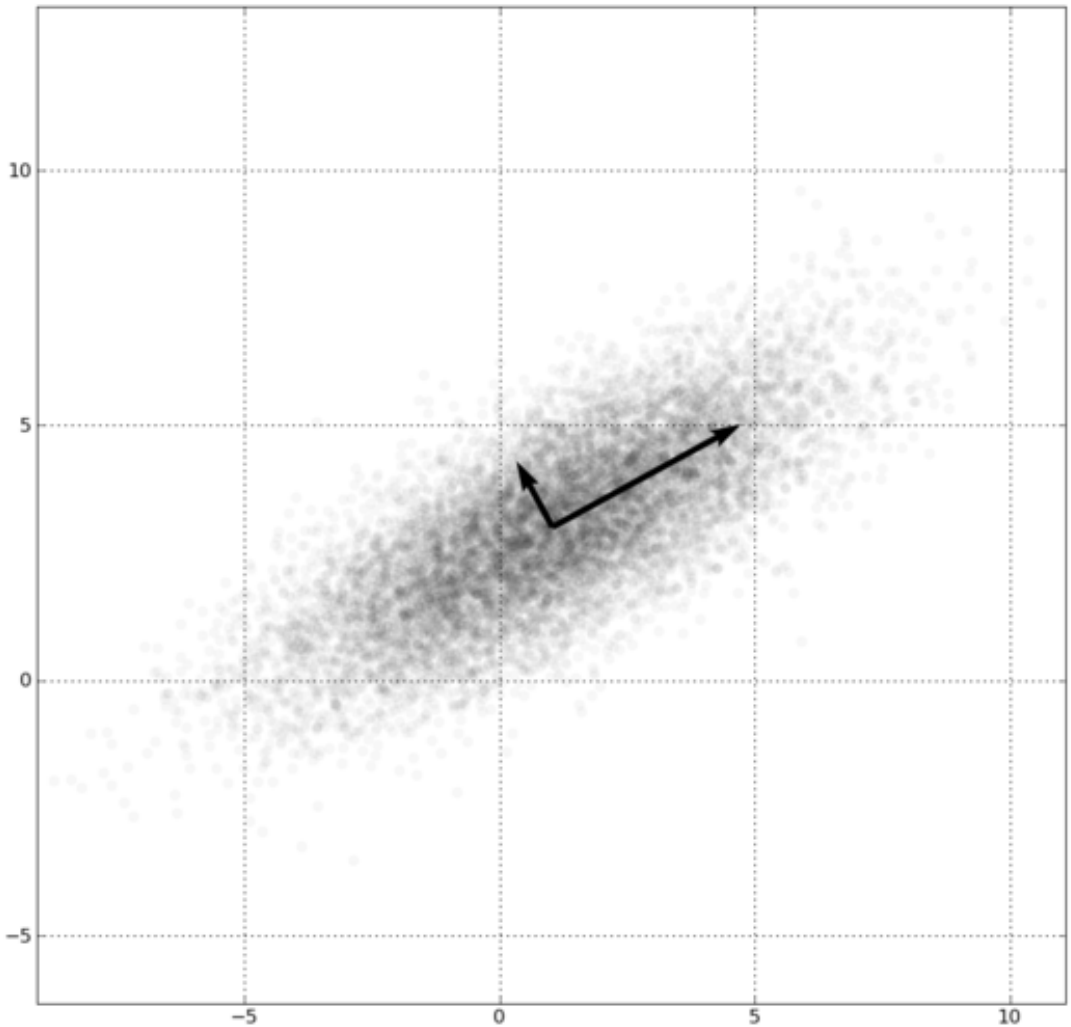




Principal Component Analysis

- **Principal Component Analysis** (PCA) is a term used for low-rank approximations in statistical analysis of data.
- Consider having m empirical data points or **observations** (e.g., daily reports) of n **variables** (e.g., stock prices), and put them in a **data matrix** $\mathbf{A} = [m \times n]$.
- Assume that each of the variables has **zero mean**, that is, the empirical mean has been subtracted out.
- It is also useful to choose the units of each variable (normalization) so that the **variance is unity**.
- We would like to find an **orthogonal transformation** of the original variables that accounts for as much of the variability of the data as possible.
- Specifically, the first principal component is the direction along which the variance of the data is largest.

PCA and Variance



PCA and SVD

- The **covariance matrix** of the data tells how correlated different pairs of variables are:

$$\mathbf{C} = \mathbf{A}^T\mathbf{A} = [n \times n]$$

- The largest eigenvalue of $\mathbf{C}$ is the direction (line) that minimizes the sum of squares of the distances from the points to the line, or equivalently, **maximizes the variance** of the data projected onto that line.
- The SVD of the data matrix is $\mathbf{A} = \mathbf{U\Sigma V}^\star$.
- The eigenvectors of $\mathbf{C}$ are in fact the columns of $\mathbf{V}$, and the eigenvalues of $\mathbf{C}$ are the squares of the singular values,

$$\mathbf{C} = \mathbf{A}^T\mathbf{A} = \mathbf{V\Sigma}\left(\mathbf{U}^\star\mathbf{U}\right)\mathbf{\Sigma V}^\star = \mathbf{V\Sigma}^2\mathbf{V}^\star.$$

Note: the singular values necessarily real since $\mathbf{C}$ is positive semi-definite.

Clustering Analysis

- Given a new data point $\mathbf{x}$, we can **project** it onto the basis formed by the principal component directions as:

$$\mathbf{V}\mathbf{y} = \mathbf{x} \quad \Rightarrow \quad \mathbf{y} = \mathbf{V}^{-1}\mathbf{x} = \mathbf{V}^{\star}\mathbf{x},$$

  which simply amounts to taking dot products $y_i = \mathbf{v}_i \cdot \mathbf{x}$.
- The first few y_i's often provide a good a **reduced-dimensionality representation** that captures most of the variance in the data.
- This is very useful for **data clustering** and analysis, as a tool to understand empirical data.
- The PCA/SVD is a linear transformation and it **cannot capture nonlinearities**.

Summary for Eigenvalues

- Eigenvalues are **well-conditioned** for **unitarily diagonalizable matrices** (includes Hermitian matrices), but ill-conditioned for nearly non-diagonalizable matrices.
- Eigenvectors are **well-conditioned** only when **eigenvalues are well-separated**.
- Eigenvalue algorithms are **always iterative**.
- Estimating **all eigenvalues and/or eigenvectors** can be done by using iterative QR factorizations, with cost $O(n^3)$.
- Iterative algorithms are used to obtain only a **few eigenvalues/vectors** for sparse matrices.
- MATLAB has high-quality implementations of sophisticated variants of these algorithms.

Summary for SVD

- The **singular value decomposition** (SVD) is an alternative to the eigenvalue decomposition that is **better for rank-defficient and ill-conditioned matrices** in general.
- Computing the SVD is **always numerically stable** for any matrix, but is typically more expensive than other decompositions.
- The SVD can be used to compute **low-rank approximations** to a matrix via the principal component analysis (PCA).
- PCA has many practical applications and usually **large sparse matrices** appear.